Inverse Kinematics

* Robot inverse kinematics problem is concerned with finding the joint variables in terms of end-effector pose.
* The objective of this problem is to find the desired joint variables that achieve a desired end-effector pose.

Problem statement:

- Given: 4x4 Homogeneous transformation matrix H represents the desired pose of the end-effector with respect to the base frame

$$^{base}H_{EE} = \begin{bmatrix} ^{0}R_n & ^{0}O_n \\ 0 & 1 \end{bmatrix}$$

- Knowing that:

$$H = {}^{0}T_n = A_1(q_1)\,A_2(q_2)\,\dots\,A_n(q_n)$$

where: $^{0}T_n$ is the general transformation matrix relating robot end-effector to the base frame.
A_i is the homogeneous transformation matrix relating link i to link $i-1$

- Required: Find $q_1, q_2 \dots q_n$ such that $H_{desired} = {}^{0}T_n(q_1, q_2, \dots, q_n)$

- Solution: For the matrices $^{0}T_n$ and H

$$T_{ij}(q_1, q_2, \dots, q_n) = h_{ij} \quad , \; i = 1,2,3 \; , \; j = 1,2,3,4$$

Thus we have 12 non-linear equations to be solved in n-variables where n is the number of joint variables (i.e. DOF)

* Difficulty of inverse Kinematics
  - The 12 nonlinear equations required to be solved are much too difficult to solve directly in a closed form.
  - Whereas the forward kinematics problem always has a unique solution that can be obtained simply by evaluating the forward equations, the inverse kinematics problem may or may not have a solution.
  - If a solution exists, it may or may not be unique.

* Approaches of finding inverse kinematics:
  - Closed form (Algebraic approaches - Geometric approach)
  - Numerical / Iterative methods
  - Machine learning approaches

* Closed form solutions
  - Closed form solutions are prefered rather than numerical solutions
  - Finding a closed form solution means finding an explicit relationship : $q_k = f_k(h_{11}, \ldots, h_{34})$, $k = 1, 2, \ldots, n$
  - Reasons for prefering closed form solutions:
    - Speed of calculation is required for rapid control of joint variables.
    - Availability of rules to choose one solution from multiple solutions regardless of the initial state of joint variables.

2b We consider 3-approaches for solving inverse kinematics
1- Geometric approach (For Simple Robot Configurations)
2- Newton-Raphson iterative method
3- learning kinematics using Neural Networks / ANFIS

Inverse Kinematics Solution
Geometric Approach

* Steps for manipulators with spherical wrist (6-DOF arms)
  1- calculate the position of wrist center
  2- Based on O_c (wrist Center) Calculate q_1, q_2, q_3
  3- From q_1, q_2, q_3 Calculate ${}^0R_3$
  4- Based on the given rotation of the end effector ${}^0R_6 \rightarrow {}^3R_6 = ({}^0R_3)^{-1} {}^0R_6$
  5- Use ${}^3R_6$ to calculate $\theta_4, \theta_5, \theta_6$ based on Euler angles rotation

* Special case for $n<6$ (example: SCARA having 4-DOF)
  • The first three steps be as above.
  4- Calculate the Rotation of EE w.r.t previous frame (i.e. θ_4) from the relation $\alpha = \theta_1 + \theta_2 - \theta_4$, ($\alpha$ is the angle of EE. w.r.t base)

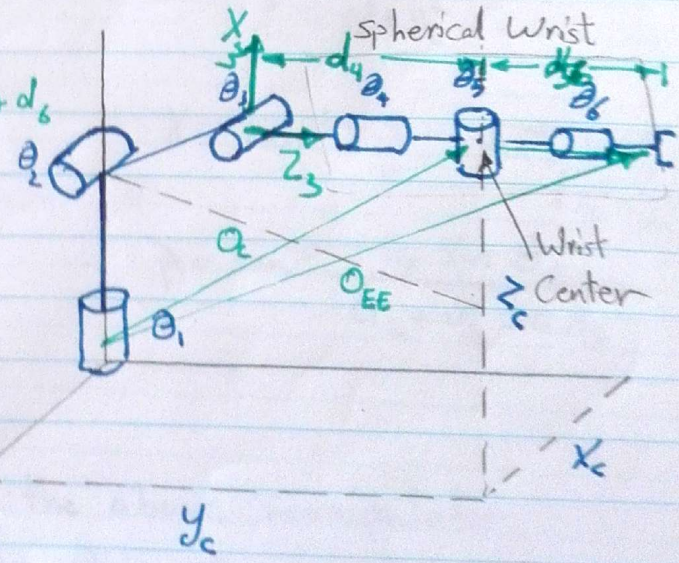
Examples

1. Articulated Manipulator:-

step1: O_c Calculation

$$O_c = O_{EE} - d_6 R_{EE} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad d_{56} = d_5 + d_6$$

a) O_{EE}: the position of EE (given)
d_6: distance in Z_6 direction, so it rotated to the base frame by R_{EE}
R_{EE}: orientation of EE (given)

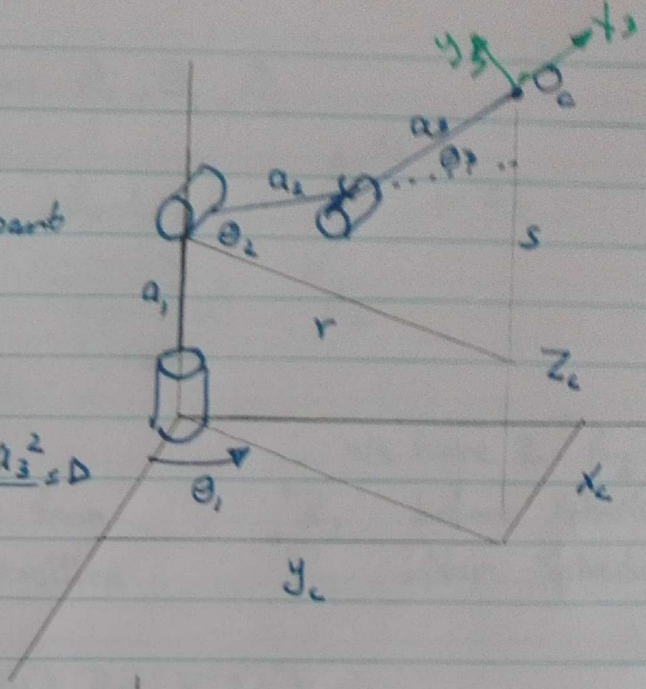


step2: $O_c = [X_c, Y_c, Z_c]^T$
use Geometry of first three links to calculate θ_1, θ_2 and θ_3

b) We use the notation $\text{atan2}(x, y) = \tan^{-1}\left(\frac{y}{x}\right)$

$$\theta_1 = \text{atan2}(x_c, y_c)$$

θ_2, θ_3 Calculate from the planner part of the arm

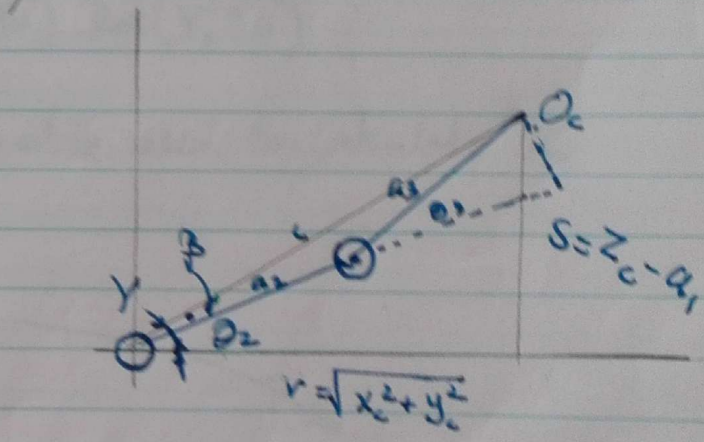


From law of Cosines:

$$\cos\theta_3 = \frac{c^2 - a_2^2 - a_3^2}{2a_2a_3} = \frac{r^2+s^2-a_2^2-a_3^2}{2a_2a_3} = D$$

$$D = \frac{(x_c^2+y_c^2)+(z_c-a_1)^2-a_2^2-a_3^2}{2a_2a_3}$$

$$\theta_3 = \text{atan2}(D, \pm\sqrt{1-D^2})$$



$$\theta_2 = \gamma - \beta$$

$$\theta_2 = \text{atan2}(\sqrt{x_c^2+y_c^2}, z_c - a_1) - \text{atan2}(a_2 + a_3\cos\theta_3, a_3\sin\theta_3)$$

$$\gamma = \tan^{-1}\frac{s}{r} = \tan^{-1}\frac{z_c - a_1}{\sqrt{x_c^2+y_c^2}}$$

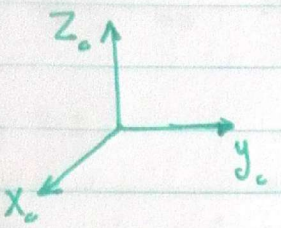
$$\beta = \tan^{-1}\frac{a_3\sin\theta_3}{a_2 + a_3\cos\theta_3}$$

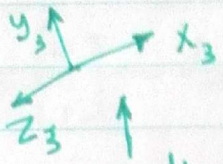
The above part is called (Inverse position)
The next part is called (Inverse Orientation)

step 3: Add spherical wrist part to the above manipulator. this will enforce you to modify θ_3: $\theta_{3new} = \theta_3 + \pi/2$ due to the difference in x_3 direction resulting from spherical wrist, to be explained below

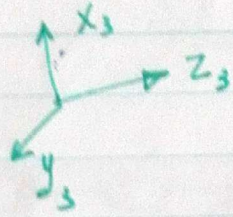
Form ${}^0R_3 = {}^0R_1\,{}^1R_2\,{}^1R_3$ based on $\theta_1, \theta_2, \theta_3$

Steps:





effect of spherical Wrist



according to $\theta_1, \theta_2, \theta_3$ calculated from Inverse position

∴ We have 2 ${}^0R_3$

${}^0R_3$ before Spherical

${}^0R_3$ after Spherical

$${}^0R_{3\,\text{after}} = {}^0R_{3\,\text{before}} * Rot(X, 90^\circ)\, Rot(Y, 90^\circ)$$

↳ the new ${}^0R_3$ is the matrix used to calculate ${}^3R_6$

step 4:

$${}^3R_6 = [{}^0R_3]^{-1}\, R_{EE}$$

Step 5:

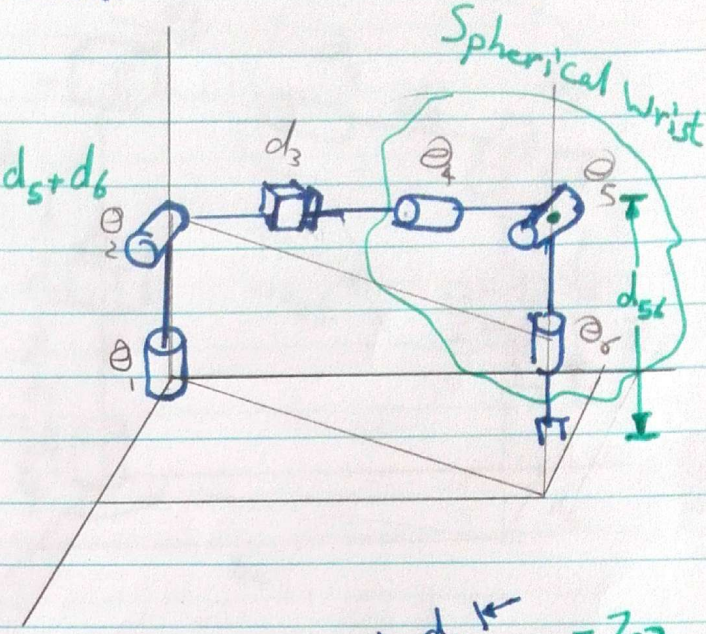
$${}^3R_6 = \begin{bmatrix} C_4C_5C_6 - S_4S_6 & -C_4C_5S_6 - S_4C_6 & C_4S_5 \\ S_4C_5C_6 + C_4S_6 & -S_4C_5S_6 + C_4C_6 & S_4S_5 \\ -S_5C_6 & S_5S_6 & C_5 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

where $S_i = \sin(\theta_i)$, $C_i = \cos(\theta_i)$

$$\theta_4 = \text{atan2}(r_{13}, r_{23})$$
$$\theta_5 = \text{atan2}(r_{33}, \sqrt{1 - r_{33}^2})$$
$$\theta_6 = \text{atan2}(-r_{31}, r_{32})$$

Euler ZYZ angles

II Spherical Manipulator with Spherical Wrist



step 1:

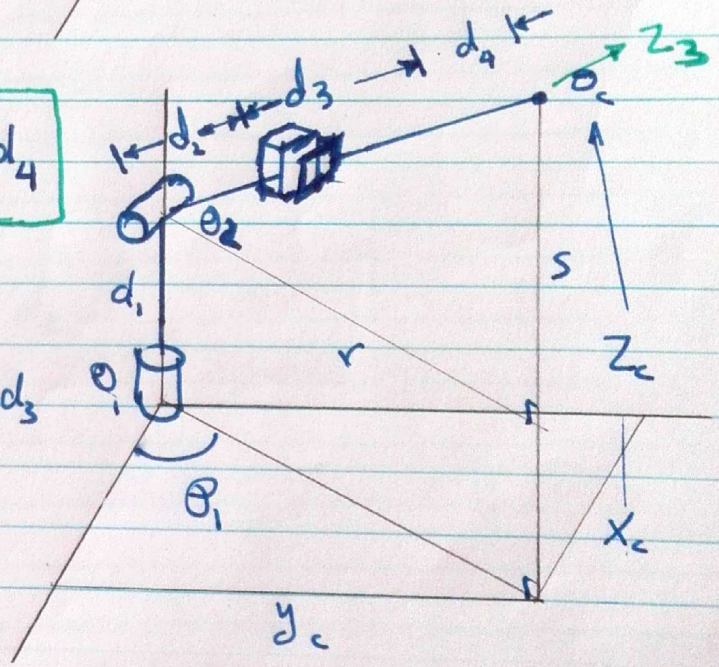
$$O_c = O_{6E} - d_{56} R_{EE} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \rightarrow d_{56} = d_5 + d_6$$

step 2:

$$\theta_1 = \text{atan2}(x_c, y_c)$$

$$\theta_2 = \text{atan2}(\sqrt{x_c^2 + y_c^2}, z_c - d_1)$$

$$d_3 = \sqrt{x_c^2 + y_c^2 + (z_c - d_1)^2} - d_2 - d_4$$



step 3:

Calculate ${}^0R_3$ based on θ_1, θ_2, d_3

step 4:

$${}^3R_6 = \left[{}^0R_3\right]^{-1} R_{EE}$$

step 5:

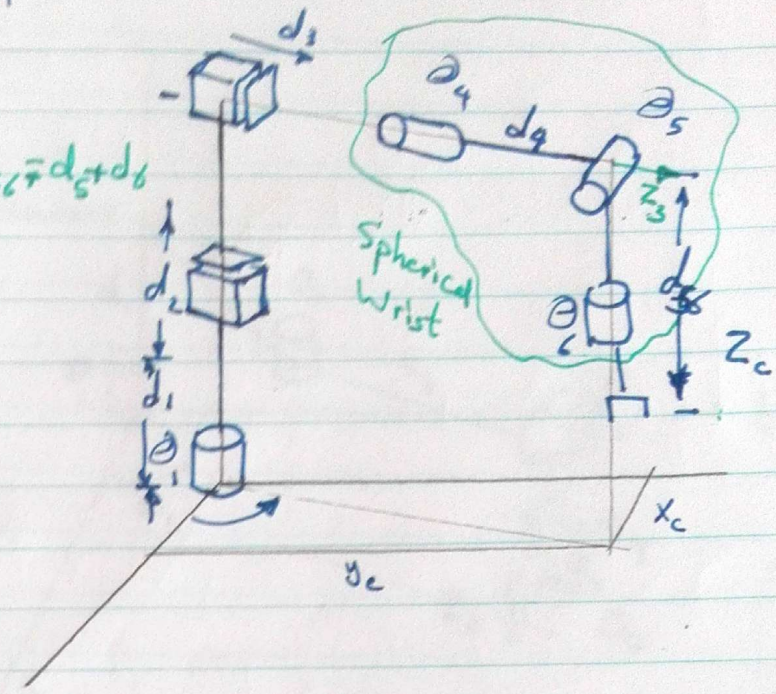
$${}^3R_6 = \begin{bmatrix} C_4C_5C_6 - S_4S_6 & -C_4C_5S_6 - S_4C_6 & C_4S_5 \\ S_4C_5C_6 + C_4S_6 & -S_4C_5S_6 + C_4C_6 & S_4S_5 \\ -S_5C_6 & S_5S_6 & C_5 \end{bmatrix}$$

$$r = \sqrt{x_c^2 + y_c^2}$$

$$s = z_c - d_1$$

$$\theta_4 = \text{atan2}(r_{33}, r_{23})$$
$$\theta_5 = \text{atan2}(r_{23}, \sqrt{1 - r_{33}^2})$$
$$\theta_6 = \text{atan2}(-r_{31}, r_{32})$$

III Cylinderical Manipulator with Spherical wrist



step 1:

$$O_c = O_{EE} - d_{6_z} R_{EE} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad d_{6_z} = d_5 + d_6$$

step 2:

$$\theta_1 = \text{atan2}(x_c, y_c)$$

$$d_2 = z_c - d_1$$

$$d_3 = \sqrt{x_c^2 + y_c^2} - d_4$$

Step 3:

Calculate ${}^0R_3$ based on θ_1, d_2, d_3

step 4:

$${}^3R_6 = [{}^0R_3]^{-1} \; R_{EE}$$

Step 5:

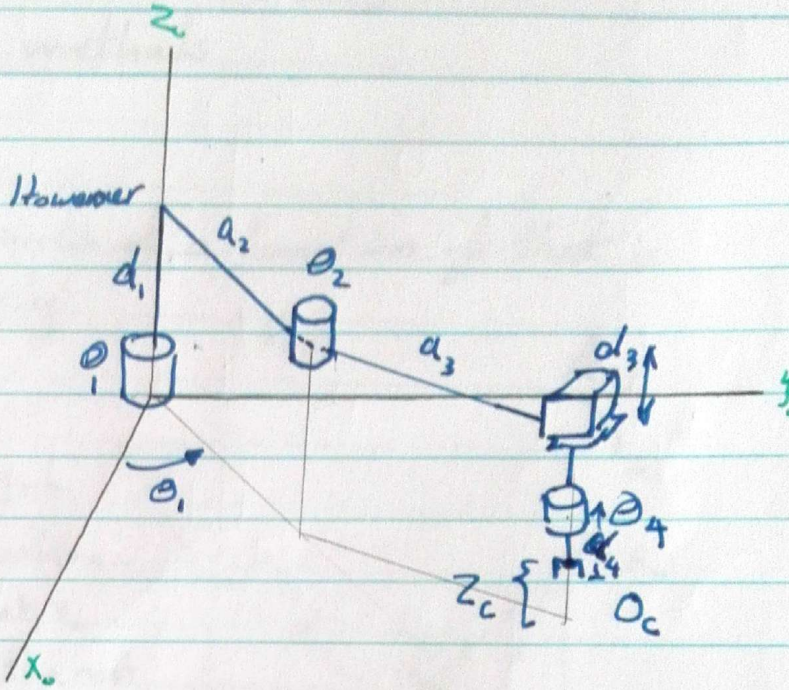
${}^3R_6$ has the form as the Previous example

$$\theta_4 = \text{atan2}(r_{13}, r_{23})$$

$$\theta_5 = \text{atan2}(r_{33}, \sqrt{1 - r_{33}^2})$$

$$\theta_6 = \text{atan2}(-r_{31}, r_{32})$$

IV SCARA Robot



step 1: In this example there is no Spherical wrist found. However a virtual wrist center can be located at the End Effector position

$O_c = O_{EE}$ (given)

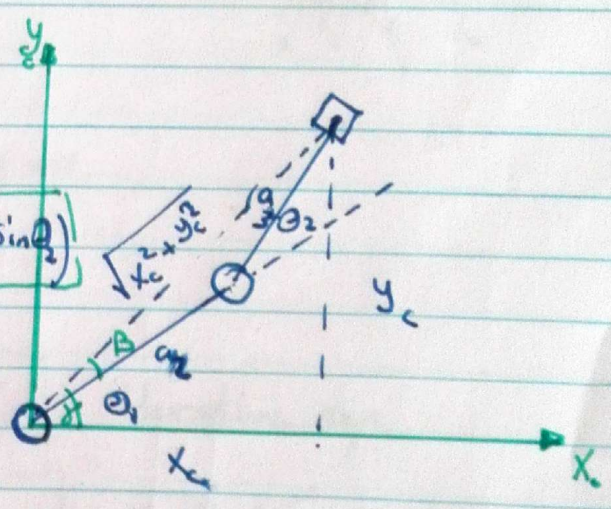
step 2:

$$\cos\theta_2 = \frac{x_c^2 + y_c^2 - a_2^2 - a_3^2}{2a_2a_3} = D$$

$$\theta_2 = \text{atan2}(D, \sqrt{1-D^2})$$

$$\theta_1 = \text{atan2}(x_c, y_c) - \text{atan2}(a_2 + a_3\cos\theta_2, a_3\sin\theta_2)$$

$$d_3 = d_1 - d_4 - Z_c$$



step 3:

Calculate ${}^0R_3$ based on θ_1, θ_2, d_3

$\theta_1 = \gamma - \beta$

$\gamma = \tan^{-1}\frac{y_c}{x_c}$

$\beta = \tan^{-1}\frac{a_3\sin\theta_2}{a_2 + a_3\cos\theta_2}$

step 4:

${}^3R_4 = [{}^0R_3]^{-1} R_{EE}$

${}^3R_4$: is a rotation about Z_3 axis (i.e $-Z_0$)

$${}^3R_4 = \begin{bmatrix} C\theta_4 & -S_4 & 0 \\ S\theta_4 & C_4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \longrightarrow \theta_4 = \text{atan2}(r_{11}, r_{21})$$

Inverse Kinematics Solution using Numerical methods

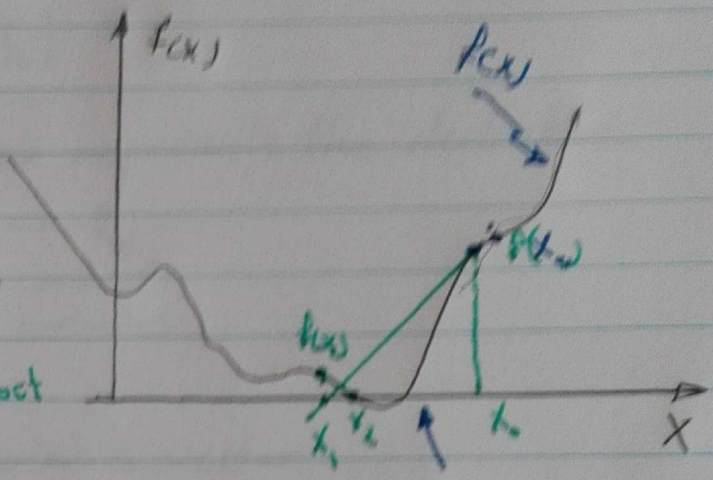
- **Newton Raphson method:**
  - Iterative method to find roots of a function f that is difficult to solve algebraicly
  - Idea:

    given: $f(x)$

    required: x such that $f(x)=0$

    solution: use tylor approximation of $f(x)$ around point x_0

    x_0: initial assumption for the root



Tylor expansion

$$f(x) = f(x_0) + f'(x_0)(x - x_0)$$

we search for x such that $f(x) = 0$

$$0 = f(x_0) + f'(x_0)(x - x_0)$$

$$x = x_0 - f(x_0)/f'(x_0)$$

Generally:

$$\boxed{x_{i+1} = x_i - f(x_i)/f'(x_i)}$$ iterative eqn

$\therefore$ We repeat the above eqn until we reach x_d that gives $\lim_{x \to x_d} f(x) = 0$

- **Use Newton Raphson method in Inverse Kinematics:**
  - Assume we given desired pose of end-effector

    $$g = [x_{ee}, y_{ee}, z_{ee}, \Phi_{ee}, \theta_{ee}, \Psi_{ee}]^T$$

  - From the forward kinematics
    - We have ${}^0T_n$ relate EE to the base ; T_n: $f(q_1, q_2, \dots q_n)$
    - form a function $f = [x, y, z, \Phi, \theta, \Psi]^T$ where each element in the vector f is a function of joint variables

- Assume $F(q_1, q_2, \dots q_n) = f(q_1, q_2, \dots q_n) - g$
  then $F = \underline{0}$ when q_i reaches desired value, $i = 1, 2, \dots n$

- We use newton raphson method to calculate q.
- From Tylor expansion of functions of Several variables

$$F(q) = F(q_0) + J(q_0)(q - q_0)$$

where

- $J(q_0)$ is the Jacobian matrix represents the derivative of $x, y, \dots \psi$ w.r.t $q_1, q_2, \dots q_n$

  ($x, y, \dots \psi$: EE pose; $q_1, q_2, \dots q_n$: Joint variables)

- $\underline{q}$ is Joint variables vector
- $\underline{q_0}$ is initial value of Joint variables vector

$$J = \begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \cdots & \frac{\partial x}{\partial q_n} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \cdots & \frac{\partial y}{\partial q_n} \\ \frac{\partial z}{\partial q_1} & \frac{\partial z}{\partial q_2} & \cdots & \frac{\partial z}{\partial q_n} \\ \frac{\partial \phi}{\partial q_1} & \frac{\partial \phi}{\partial q_2} & \cdots & \frac{\partial \phi}{\partial q_n} \\ \frac{\partial \theta}{\partial q_1} & \frac{\partial \theta}{\partial q_2} & \cdots & \frac{\partial \theta}{\partial q_n} \\ \frac{\partial \psi}{\partial q_1} & \frac{\partial \psi}{\partial q_2} & \cdots & \frac{\partial \psi}{\partial q_n} \end{bmatrix}$$

Here $x, y, z, \phi, \theta, \psi$ are the functions representing the pose of end-effector in terms of joint variables

- The objective is finding q such that $F(q) = 0$

$$\therefore \boxed{q = q_0 - J^{\#}(q_0) F(q_0)}$$ Iterative eqn

where

$J^{\#}$ is Pseudo inverse of non-square Jacobian matrix

Inverse Kinematics Solution using Machine Learning

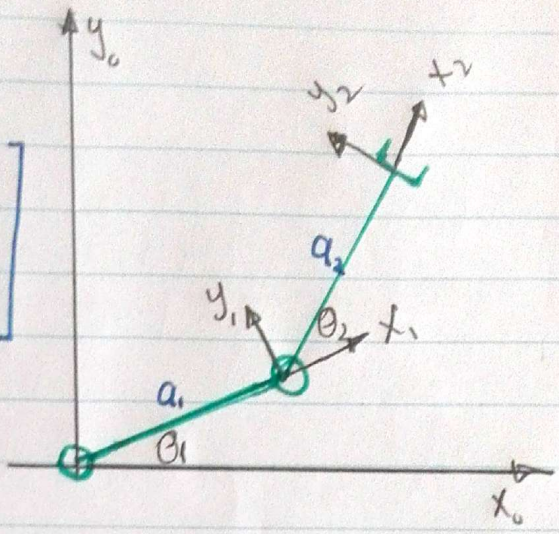
- **Nonlinear functions representation**
  - One application of computational intelligence models (Inferences) such that Fuzzy Systems, Neural Networks, and ANFIS is to model systems described by nonlinear functions.
  - Parameters of these models are adjusted using machine learning techniques

- **Use of NN/ANFIS to model inverse kinematics.**
  - Forward kinematics equations can be derived simply using DH-convention
  - From the deduced ${}^{0}T_{n}(q_1, q_2, \dots q_n)$ we can form a function $f(q_1, q_2, \dots q_n) = [x_{ee}, y_{ee}, z_{ee}, \phi_{ee}, \theta_{ee}, \psi_{ee}]^T$. the elements of vector f are the end effector pose in terms of q's
  - Create a NN/ANFIS model with end-effector pose as inputs and joint variables as outputs.
  - Create a dataset by applying different values of qs and drive the corresponding pose.
  - The data set created must be adjusted to make pose as input and q's as labeled data (training output)
  - Using dataset apply a machine learning algorithm to adjust weights of NN/ANFIS
  - After training step the deduced inference with adjusted weights can be used on-line to calculate suitable q's for desired pose.

Example: Two link Planner Manipulator inverse Kinematics

step 1: Calculate Forward Kinematics

$$H_{EE} = \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) & 0 & d_x \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & 0 & d_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

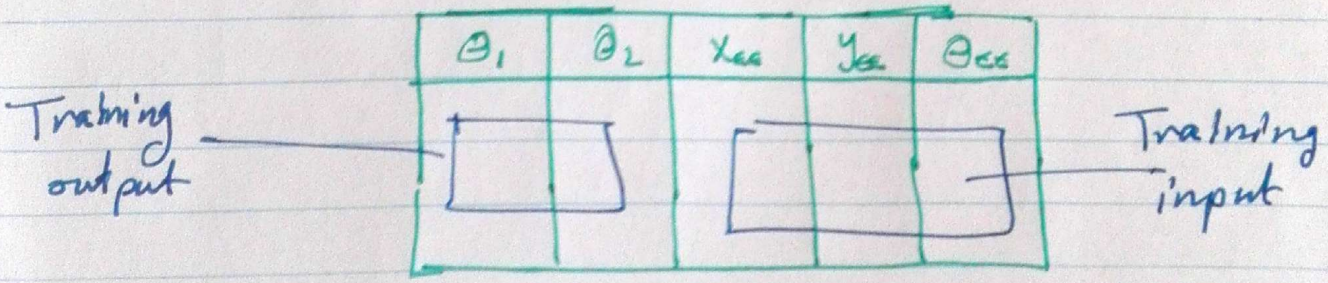


$$d_x = a_1\cos(\theta_1) + a_2\cos(\theta_1+\theta_2)$$
$$d_y = a_1\sin(\theta_1) + a_2\sin(\theta_1+\theta_2)$$

step 2: Construct $f(q) = [x_{EE}, y_{EE}, \theta_{EE}]^T$

$$\therefore \quad f(q) = \begin{bmatrix} a_1\cos(\theta_1) + a_2\cos(\theta_1+\theta_2) \\ a_1\sin(\theta_1) + a_2\sin(\theta_1+\theta_2) \\ \theta_1+\theta_2 \end{bmatrix}$$

step 3: Apply different values of θ_1 and θ_2 and find corresponding $f(q)$ to form dataset



step 4: Construct a NN/ANFIS model with $[x_{EE}, y_{EE}, \theta_{EE}]$ as inputs and $[\theta_1, \theta_2]$ as outputs

step 5: Apply a machine learning technique (Back Propagation algorithm) using the dataset to adjust model parameters.